THE AGILE PLANNER: CONFIGURABLE C2 ON THE MOVE

John Shaw, Alberto Russo, Ron James, Paul Calnan, Michael Sao Pedro and David Pepyne
BAE Systems Advanced Information Technologies
6 New England Executive Park
Burlington, MA 01803

ABSTRACT

Current Army C2 systems cannot integrate "come as you are" capabilities into operations making it difficult to combine participants into a cohesive force. This paper describes the Agile Planner concept developed by BAE-AIT as robust approach for configurable C2 "on the fly". With an Agile Planner, participants declare their capabilities when they register into the system. Once they do so, an Agile Planner can assemble a system architecture from these declared capabilities and determine how to employ the participants to accomplish a mission. The Agile Planner concept is illustrated with a case study showing its ability to robustly build Missile Defense from "come as you are" assets.

1. INTRODUCTION

Today's Command, Control, Battle Management and Communications (C2BMC) systems are hard coded to today's architecture and tactics. Integrating new defensive elements requires a lengthy development cycle, providing the enemy sufficient lead time to discover and defeat our capabilities. Additionally, current C2BMC systems cannot integrate "come as you are" capabilities into operations. Platforms and units invariably arrive on the scene with different versions of mission systems, complicating the effort to combine participants into a cohesive force.

Ideally, C2BMC systems would automatically identify the capabilities that participants actually possess and determine the true interoperability that could be achieved among them. This would let participants declare their capabilities when they arrive on the scene. Such systems would thereafter assemble the de-facto system architecture from these declared capabilities and would determine how to employ the participants to accomplish the mission.

BAE Systems Advanced Information Technologies (BAE-AIT) has developed the *Agile Planner*, a planning and execution control capability to assemble agile C2BMC systems. A key innovation of the Agile Planner is its ability to express the behavior, operating rules, and constraints for a target defense architecture as Extensible Markup Language (XML) profiles which the Agile Planner accepts at runtime as architecture profile "plug-ins." The XML schema provides a common description of the defense system behavior, rules, and constraints, thus permitting interoperability between various system elements. Designers and operators can create new XML profiles, or extend existing profiles, for new defense architectures and engagement rules without having to write any new software.

A second innovation of the Agile Planer is a Dynamic Constraint Reasoning system that provides both run-time constraint declaration and solution. This system performs "on-the-fly" translation of objectives and participant capabilities into a network of coupled constraints. Constraint-Based Interval Planning (CBIP) algorithms, originating with machine learning planning systems, determine if the constraints have feasible solutions. The result is a system where resource management objectives are expressed in an extensible ontology that is transformed at run time into constraint networks whose solution is a feasible coordinated plan.

This paper presents the Agile Planner's innovative problem formulation mechanism, as well as its architecture and algorithms. Additionally, we provide a case study using the Agile Planner as the backbone of the Agile Engagement Planner, a missile defense engagement planning system developed by BAE Systems AIT.

2. THE AGILE PLANNER

The Agile Planning vision is a system able to discover in real-time the capabilities that different resources offer, the capabilities that mission objectives require, and then to daisy chain the offered capabilities at run-time into a cohesive unit of action that possesses the range of capabilities required by the mission. Abstracted to the level of capabilities and constraints, an Agile Planner is able to integrate "come as you are" participants into a cohesive force.

2.1 Architectural Overview

Fig. 1 shows the basic architecture of an Agile Planner.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
01 NOV 2006	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
The Agile Planner: Configurable C2 On The Move	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **BAE Systems Advanced Information Technologies 6 New England Executive Park Burlington, MA 01803**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
See also ADM002075., The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**UU**	**8**	

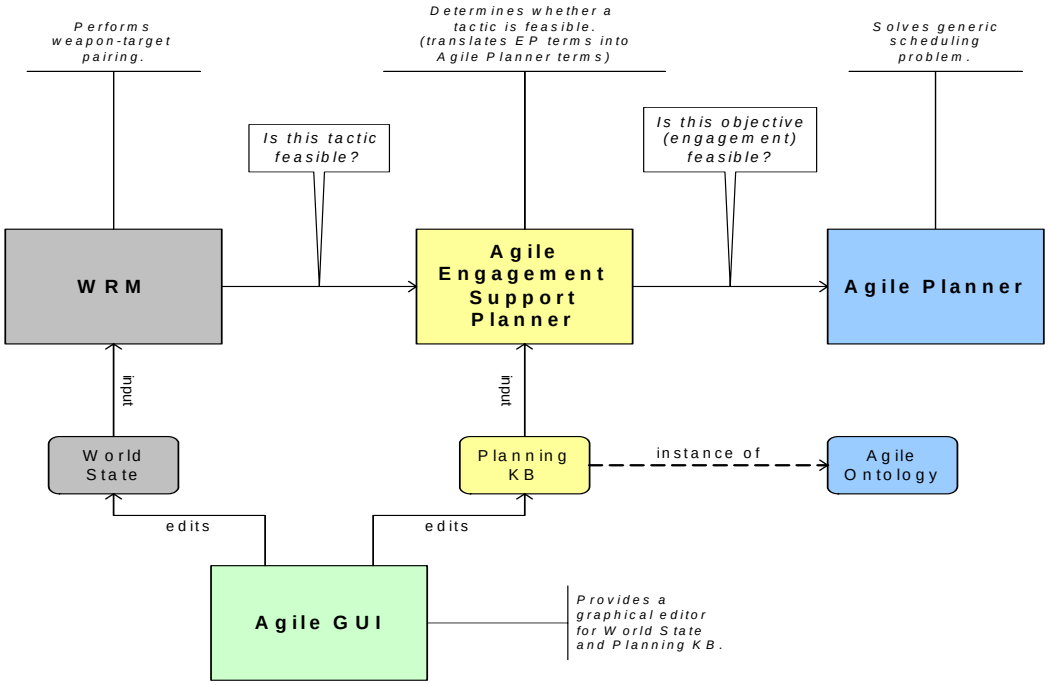

Fig. 1 Architecture of an Agile Planner.

The Agile Planner Graphical User Interface (GUI): Users create and configure weapon system architectures and deployment rules using the Agile Planner Graphical User Interface (Agile GUI). A weapon system's architecture describes how the weapon system behaves, the activities that must be carried out to employ the weapon, the performance thresholds those activities must meet, and so on. Engagement constraints can be temporal (such as the time required to carry out a support task), resource-oriented (tied to the attributes of the resource), or spatial (such as the definition of no fly zones).

The Agile GUI serves as an editor for two XML-based input file formats: the World State and the Planning Knowledge Base (PKB). The World State is the input for the Weapons Resource Manager (WRM) (described below). It specifies various runtime configuration options for the WRM. The PKB contains the information required by the AESP and the AP to solve the weapon to target scheduling objective. This includes the activities and activity types, resources and resource types, constraints, and Course of Action (COA) templates that are required to select a weapon platform and its support requirements.

The Weapon Resource Manager (WRM): The WRM performs weapon planning activities and is specific to the problem domain under consideration. It ingests commander's guidance, real-time system status and reports of hostile activities, and prepares prospective weapon-target pairing options. A prospective option matches a weapon platform to a target and specifies a desired engagement window. The Weapon Planner passes each prospective weapon-target pairing option to the AESP component.

The Agile Engagement Support Planner (AESP): The AESP works as an intermediary between the WRM and the AP. The WRM operates on its own data structures and concepts which are specific to problem domain under consideration. The AP, however, is a general-purpose scheduling tool that operates on data structures and concepts generic to the activity scheduling domain. The AESP wraps the AP by accepting as input a specific problem formulation and translating it into the generic problem formulation required by AP.

The Agile Planner (AP): The core of the agile architecture is the AP. The AP obtains the basic problem specification from the XML-based PKB. The AP operates in terms of general-purpose activity scheduling concepts to determine whether there is at least one feasible way to carry out all of the required support activities that must be performed to achieve each desired weapon to target pairing option.

2.2 Agile Ontology

Agile Planning is model-driven by models expressed in an Agile Ontology. An Agile Ontology is an abstract

language for describing the elements of the C2 resource management problem under consideration. Our Agile Ontology encompasses *Objectives*, *Activities*, *Capabilities*, and *Resources*. An *Objective* is a mission goal that the Agile Planner will try to accomplish. Objectives are the input to the AP. An *Activity* is a task or a collection of tasks that must be carried out to accomplish an objective. *Resources*, which can be reusable (e.g., aircraft, radars) or consumable (e.g., munitions, fuel), are entities that posses the *capability* to perform various activities. *Capabilities* are domain dependent and include such notions as flight capability in the air planning domain, intercept capability in the missile defense domain, or transport capability in the logistics domain. Using the language of our Agile Ontology, users can assemble "on-the-fly" XML profiles for mission objectives and available resources, which the Agile Planner can immediately integrate into its planning algorithms – all without changing a single line of the software.

2.3 Dynamic Constraint Reasoning

Armed with a given set of mission objectives, Dynamic Constraint Reasoning algorithms seek to generate plans for executing them. Central to these algorithms are *Course of Action (COA) Templates*. These describe how to achieve one or more objectives. A COA lists the activities that must be performed and the constraints that must hold between them. Activities can be constrained by resources, e.g., requiring that two activities be performed by the same resource or requiring a resource have a specific range capability. Activities can also be constrained temporally, to enforce a particular ordering between them. A *Resource Option* specifies a candidate resource that can be used to carry out a given activity. Each activity has one or more resource options associated with it. Generating a plan involves selecting resources and scheduling their activities.

Planning in an Agile Planner is performed in a "Reasoner Pipeline," see Fig. 2. This pipeline is composed of a sequence of planning components which collaborate to formulate a complete plan. The pipeline follows a "Chain of Responsibility" design pattern. Each element in the pipeline solves a portion of the scheduling problem. Taken together, the entire pipeline solves the entire scheduling problem.

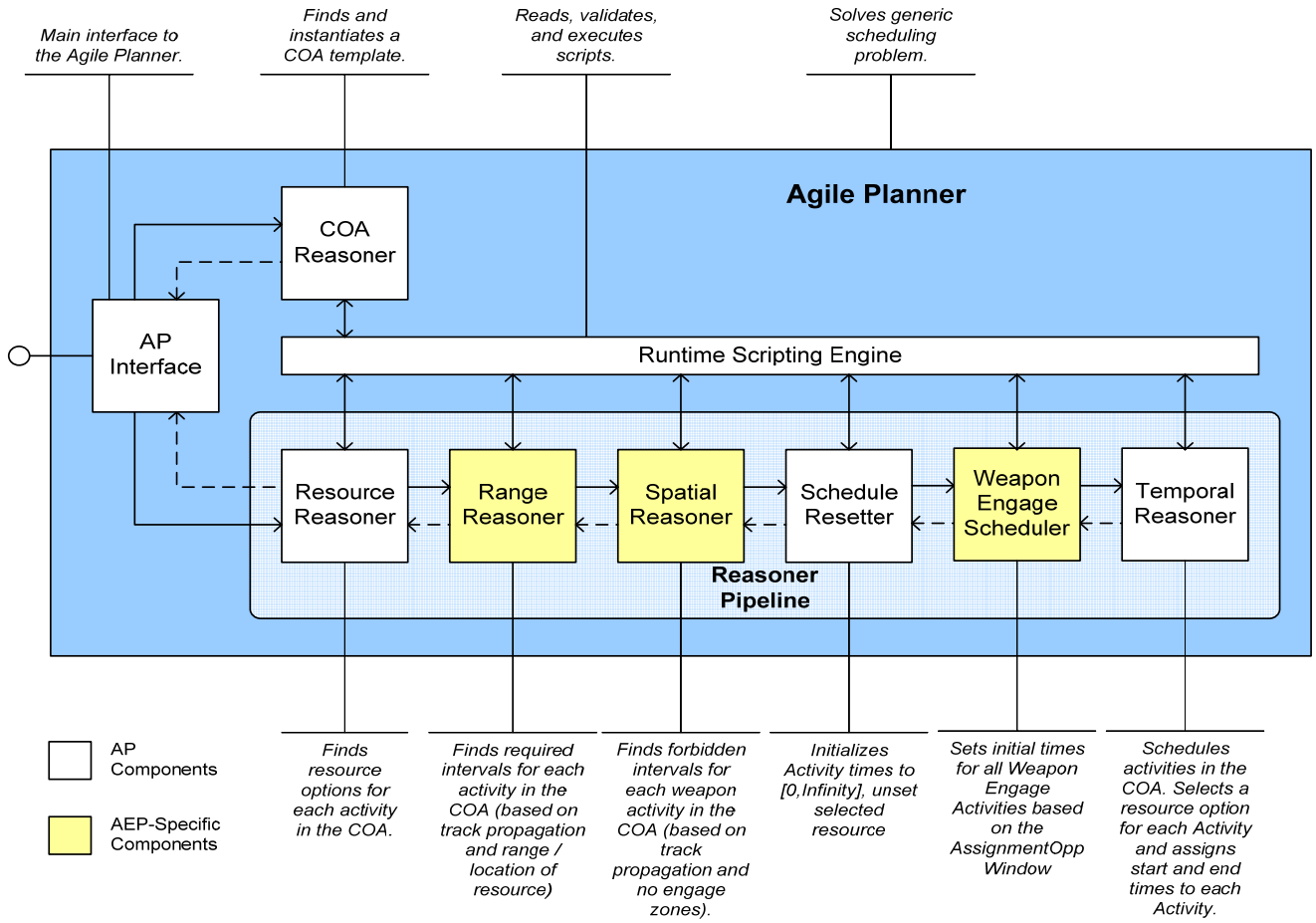

Fig. 2 The Agile Planner Reasoner Pipeline.

The *Agile Planner (AP) Interface* is the entry point to the Agile Planner pipeline. Its job is to coordinate the various components of the pipeline. It takes as input objectives and a plan knowledge base consisting of a set of course of action (COA) templates describing how to accomplish different kinds of objectives. The *COA Reasoner* examines the set of COA templates to determine which are applicable to each specified objective. If the objective's required capabilities match the capabilities provided by the COA, the pipeline is invoked to determine if the constraints (resource, spatial, temporal) can be satisfied. The *Architecture Profile Reasoner* identifies the support activities and constraints contained in the weapon's XML profile. It uses this information to construct an activity graph containing the engagement support activities, activity deadlines, precedence constraints, and performance thresholds required to support the engagement. The *Resource Reasoner* determines which assets available to the system are prospectively able to support each activity. The *Range Reasoner* determines whether each asset selected by the Resource Reasoner is within range during the required time window. If a resource is never within range, it is removed from consideration. The *Spatial Reasoner* then adjusts the time windows of each activity to ensure that it occurs (or does not occur) within a specific region of space. Constraint-Based Interval Planning algorithms in the *Temporal Reasoner* determine if there is at least one way to carry out all of the support activities using the resource options chosen above and meeting the constraints contained in the architecture profile and generated by the other reasoning elements

Since the Agile Planner architecture is designed to be flexible, the reasoner pipeline allows external problem specific planning components to be included in its execution flow. In Fig. 2 the modules shown in white are domain independent algorithms while the modules shown in grey are specific to the ballistic missile engagement problem, to be presented as a use case later in this paper.

3. CONSTRAINT-BASED PLANNING

The actual planning within an Agile Planner is done using constraint-based planning algorithms. These algorithms, which originated in the machine learning community (Dechter, Meire, Pearl, 1991; Dechter, 2003), perform a run-time transformation from problem specific data structures to data structures and concepts generic to the activity scheduling domain and then determine if there is at least one feasible way to carry out all of the activities specified in the set of COAs. The general goal is to maximize the number of mission objectives that can be accomplished within the constraints of the available resources.

Fig. 3 shows the general structure of the constraint-based scheduling paradigm used within the Agile Planner. The scheduling paradigm combines search over the space of resource options with constraint propagation techniques to evaluate the consequences of particular assignments of resource options to activities. Each search step attempts to assign a resource option to an activity or impose an ordering between activities. Constraint propagation guides the search by detecting infeasible choices and by narrowing future choices.

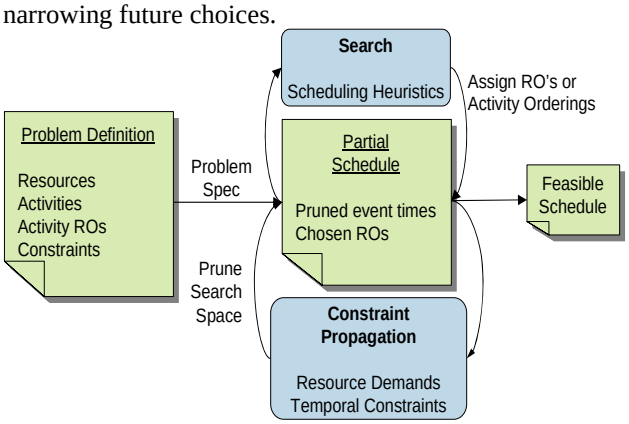

Fig. 3. Constraint-Based Scheduling Paradigm.

The Agile Planner constraint propagation engine integrates state-of-the-art precedence graph techniques for handling resource constraints (Laborie, 1993) with simple temporal problem (STP) graph techniques for handling temporal constraints. Combined, these graphical structures capture resource usage and activities over time and global constraints between them. Constraint propagation on these graphical structures is done in two steps. The first step attempts to balance constraints over the set of precedence graphs. If there is no solution that balances the constraints, then this reveals one or more inconsistencies due to resource usage violations that are inherent with the current search choice. The Agile Planner backtracks to locate an alternative resource option. The second step runs arc consistency on a STP graph to check for temporal constraint violations. Temporal violations that cannot be repaired by time window adjustments require backtracking to try another resource option.

Several variants of the dynamic constraint solver have been developed. In one variant, used in an application for generating aircraft flight plans involving aerial refueling and escort jamming for suppression of enemy air defense, all of the aircraft to target options are scheduled at the same time using coupled constraint satisfaction techniques. In the ballistic missile defense application to be described in the next section, the interceptor to ballistic missile options are ordered by maximal marginal return and scheduled sequentially in that order.

4. CASE STUDY: THE AGILE ENGAGEMENT PLANNER

Under contract to the Missile Defense Agency, BAE-AIT developed the Agile Engagement Planner (AEP) to prove the concept of the Agile Planning approach. In this section we use the AEP to illustrate some of Agile Planning's key advantages by demonstrating the ease with which new interceptor technologies can be incorporated and integrated with existing ones.

4.1 Ballistic Missile Defense Engagement

Broadly speaking, the ballistic missile engagement problem involves assigning ballistic missile interceptors to incoming ballistic missiles. Each such interceptor to missile assignment option requires certain support such as sensors for detecting and tracking the incoming missiles and communication uplinks for sending guidance commands. The goal is to determine a support plan that maximizes the number of successful engagements within the constraints of the available interceptor batteries and support resources.

Fig. 4 shows a typical missile defense engagement sequence. Once deployed, a warhead (i.e., a Re-entry Vehicle or RV) may be surrounded by target cloud of decoys and debris. If we are using a kinetic weapon to engage the RV, we first need to fly an interceptor out to the target cloud. Sometime before intercept, the interceptor needs to pick out the RV from the other objects in the cloud, home in on it, and either hit it directly (for a hit-to-kill weapon), or pass within an acceptable blast distance of the target (for a fragmentary blast weapon). The interceptor may locate the RV by its own means, or the RV may be designated by an off-board source.

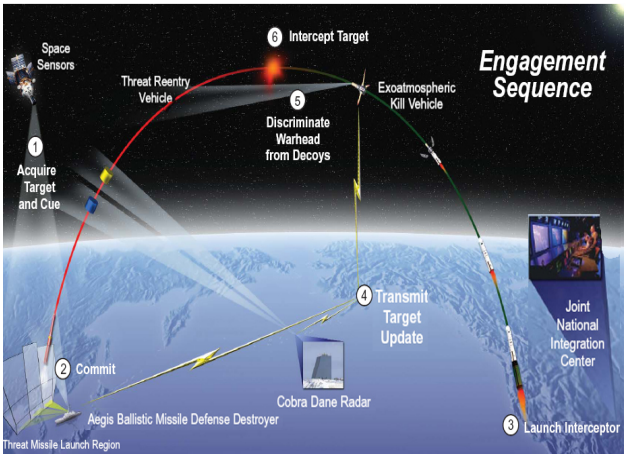

Fig. 4 Missile Defense Engagement Sequence.

4.1 Defining Capabilities

Central to an Agile Planner is an ontology for describing in a common way through XML profiles the "come as you are" capabilities and constraints of the available participants. These XML weapon system architecture profiles come as input files or can be entered in real-time by an operator at the AP GUI. In what follows, we will give two examples to illustrate the ease with which different system architectures can be assembled into an integrated cohesive defensive force.

For our first weapon system, let us consider a relatively "low-cost" interceptor. As shown in Fig. 5, this interceptor is unable to locate an RV on its own. It requires In Flight Target Updates (IFTUs) to guide it to the target cloud during flight. It also requires a Threat Object Map (TOM) to assist it in picking out the RV within the cloud. This interceptor requires support in the form of sensing resources (e.g., a radar) to generate the IFTUs and TOM, and communication resources to forward the data to the interceptor. Sensing resources are also needed for kill assessment after intercept.

To configure AEP to use this weapon system architecture, we need to define its architecture and its engagement rules. This is done in real-time by an operator through the AEP GUI. First, AEP must be made aware of the required support activities. From the description above, we can derive seven support activities that must be performed: 1) generate coarse IFTU, 2) send coarse IFTU, 3) generate fine IFTU, 4) send fine IFTU, 5) generate TOM, 6) send TOM, and 7) kill assessment.

Each of these support activities requires a certain capability. A sensing capability is required to generate the IFTUs and TOM and to perform kill assessment. Communications capability is required to send the IFTUs and TOM to guide the interceptor. Thus, we define two capabilities: sensing and communications.

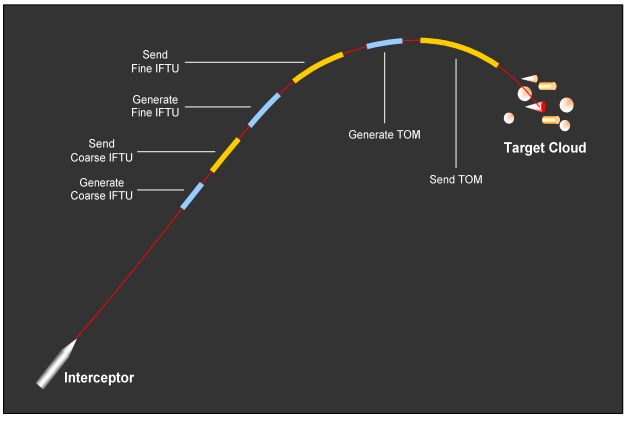

Fig. 5. The engagement sequence for our initial weapon system architecture

Next, we define the resources that provide the capabilities specified above. We can define an In-Flight

Interceptor Communications System (IFICS) resource that provides communications capabilities. We can also define a Ground Based Radar (GBR) resource to provide the sensing capabilities. In addition to capabilities, resources have a location, a range, and either an inventory (for expendable resources) or a capacity (for reusable resources). Since both the IFICS and the GBR are reusable resources, their capacities are defined at 100%.

Lastly, we define an architecture profile to specify the order the support activities need to be performed. Fig. 6 shows the architectural profile for our low cost interceptor.

4.2 Integrating New Capabilities

Once the Agile Engagement Planner has been made aware of a weapon system architecture it is able to use it in constructing engagement plans. As other interceptors, sensors and communications resources become available they can be added to the AEP system via the GUI in exactly the same way as above. By expressing their capabilities and constraints in a common way, the AEP can utilize those new resources in future engagement plans. In this way, weapon systems that had never previously been used together can be combined into a single integrated system.

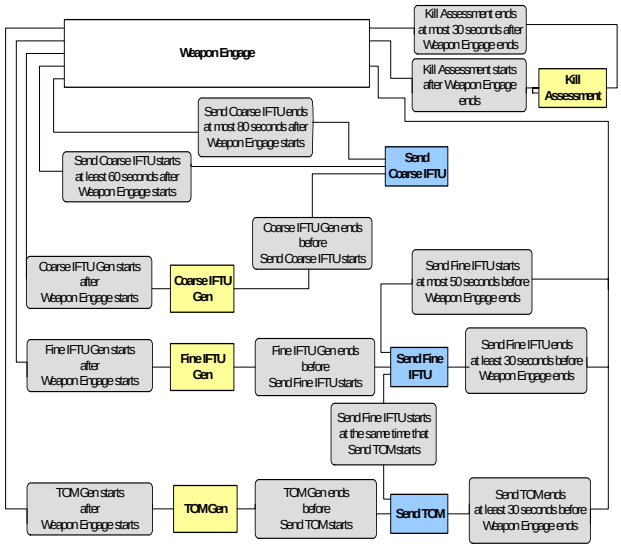

Fig. 6 Architectural Profile for a "low cost" interceptor.

To illustrate let us introduce a new interceptor that uses a completely new kind of support – a space based laser illuminator for target designation. That is, as shown in Fig. 7, this interceptor works by flying out to a warhead acquisition location. Upon arrival, a space-based laser designator locates and illuminates the RV. The interceptor looks for the laser reflection from the RV, locks on to it, and steers itself to intercept.

To add a new architecture such as this one to today's missile defense engagement planning systems would require a long development process. Branching conditions would need to be added to the engagement code; as more weapon architectures are added, the complexity of the code would increase. Likewise, every time a new architecture is added, the code would need to be recertified to ensure that it functions properly when deployed.

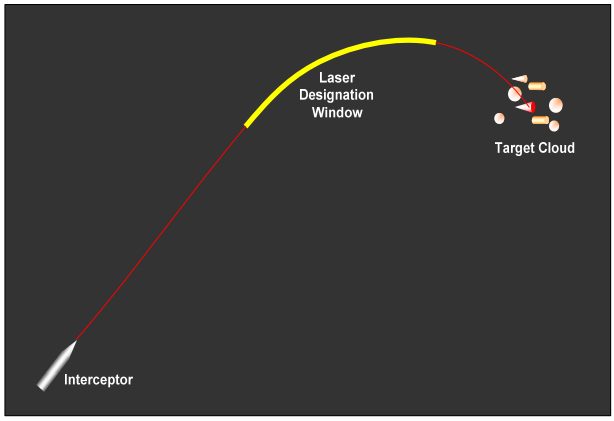

Fig. 7 Engagement Sequence for an "advanced" weapon system.

With Agile EP all we need to do is define the support activities required by this new weapon system architecture. In particular, we need to define two support activities: illuminate and kill assessment. Each of these support activities requires a resource to provide a certain capability. The kill assessment activity requires a sensing capability. The illuminate activity requires an illumination capability. Since the sensing capability was defined in our previous example, we do not need to redefine it. We only need to define the illumination capability to support this new weapon system architecture.

Next, we define resources that provide the capabilities specified above. Again, we are able to reuse the radar resource defined in the previous example. We can define a new Laser Illuminator resource and provide it with a location.

We also need to define the new interceptor type. Using the AEP GUI, we are able to define weapon parameters that are used by the WRM when computing engagement plans. We then need to define a weapon site that has this new weapon type in its inventory.

Finally, we define a new architecture profile that specifies when the illuminate and kill assessment

activities must occur relative to the weapon engage. This architecture profile is shown in Fig. 8.

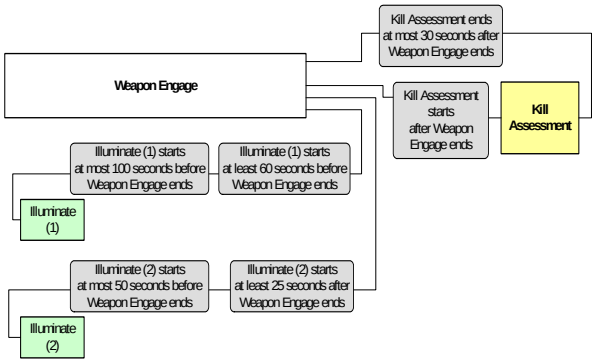

Fig. 8 Architectural Profile for an "advanced" interceptor.

4.3. Constraint Reasoning in the AEP

Once the available resources, their support requirements, and their capabilities have been defined, engagement problems can now be defined and solved. Referring back to Fig. 1, the problem to be solved begins with the Weapons Resource Manager (WRM), which generates a list of weapon to target options. In AEP, these weapon to target options are scored by marginal return, and AEP schedules weapon to target options to achieve maximal marginal return. This involves selecting appropriate interceptors and scheduling all of the required support activities necessary to carry out the engagement.

Solution involves a process of "capability matching" via the reasoner pipeline in Fig. 2. Activities require capabilities, e.g., the "generate coarse IFTU" activity requires a resource that provides the sensing capability. Rather than looking for a particular kind of radar to perform this activity, AP looks for a resource that provides the sensing capability. After finding such a resource, the AP then ensures that the resource meets the constraints of the mission. This requires de-conflicting three main categories of constraints:

Temporal Constraints: Temporal constraints can be expressed in natural language via a GUI editor. They translate into a superset of Allen relationships, e.g.,

- Illuminate starts at most <TBS> seconds before Weapon Engage ends.
- Illuminate ends at most <TBS> seconds before Weapon Engage ends.
- Kill Assessment ends at most <TBS> seconds after Weapon Engage ends.
- Kill Assessment starts after Weapon Engage ends.

TBS = To Be Specified. This is a user specified time that depends on weapon system used.

Resource Constraints: Each activity requires a resource with a given capability. The planner is free to choose any resource that meets the specified capability. Resources have constraints that limit their capabilities. The constraints might be static, e.g., a radar may have limited range and angular resolution accuracy. The constraints might be dynamic, .e.g., remaining inventory of interceptors or current range to target.

Spatial Constraints: Activities might be constrained to meet spatial constraints, e.g., engagement may have to avoid "no-engage" zones or may have to occur within a certain distance of the target.

Using a case study from the ballistic missile defense (BMD) domain, we illustrated several of the key advantages of the Agile Planning architecture. First, we showed how through modifications to AEP's input files we are able to define an entirely new BMD system on the fly. AEP can now seamlessly utilize this new weapon system to construct its engagement plans. There was no need to create models to simulate new weapon systems, radars or communications systems. There was no need to write or debug any new code. Weapon systems that in the past were separately planned can now be integrated into a single cohesive force.

5. CONCLUSION

This paper presented Agile Planning as a generic and flexible approach for a wide range of complex resource management problems. Agile Planning solves problems that require scheduling activities to use resources that provide certain capabilities. Uses range from resource assignment and scheduling problems in areas from project management to air campaign planning, to logistics, to missile defense. Where today's defense planners often implement these assignment and scheduling calculations using hard-coded software specific to today's weapons and tactics, the innovation of Agile Planning is its use of a ontological abstraction that models resource assignment problems in general terms of capabilities and constraints, and a dynamic constraint reasoning mechanism to determine on the fly what calculations need to be performed and thereafter solve them. The end result is a robust planning approach able to integrate existing capabilities and evolve with new ones.

REFERENCES

Laborie, P., 2003: "Algorithms for Propagating Resource Constraints in AI Planning and Scheduling: Existing

Approaches and New Results," *Artificial Intelligence*, Vol. 143, pp. 151-188.

Dechter, R., Meiri, I. and Pearl, J., 1991: "Temporal Constraint Networks," *Artificial Intelligence*, Vol. 49, pp. 61-95.

Dechter, R., 2003: *Constraint Processing*, Morgan Kaufmann.